Yee & Associates, P.C.

4100 Alpha Road
Suite 1100
Dallas, Texas 75244

Main No. (972) 385-8777
Facsimile (972) 385-7766

Facsimile Cover Sheet

To: Commissioner for Patents for **Examiner Anh Ly** **Group Art Unit 2162**	Facsimile No.: **703/872-9306**
From: Michele Morrow Legal Assistant to Francis Lammes	No. of Pages Including Cover Sheet: **42**
Message: Enclosed herewith: • Transmittal Document; and • Appeal Brief.	
Re: Application No. **09/544,274** Attorney Docket No: **AUS000185US1**	
Date: Thursday, June 23, 2005	
Please contact us at (972) 385-8777 if you do not receive all pages indicated above or experience any difficulty in receiving this facsimile.	*This Facsimile is intended only for the use of the addressee and, if the addressee is a client or their agent, contains privileged and confidential information. If you are not the intended recipient of this facsimile, you have received this facsimile inadvertently and in error. Any review, dissemination, distribution, or copying is strictly prohibited. If you received this facsimile in error, please notify us by telephone and return the facsimile to us immediately.*

PLEASE CONFIRM RECEIPT OF THIS TRANSMISSION BY FAXING A CONFIRMATION TO 972-385-7766.

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: **George**

Serial No.: **09/544,274**

Filed: **April 6, 2000**

For: **Apparatus and Method for Deletion of Objects from an Object-Relational System in a Customizable and Database Independent Manner**

35525
PATENT TRADEMARK OFFICE
CUSTOMER NUMBER

§ Group Art Unit: **2162**
§
§ Examiner: **Ly, Anh**
§
§ Attorney Docket No.: **AUS000185US1**
§
§
§
§
§

Certificate of Transmission Under 37 C.F.R. § 1.8(a)
I hereby certify this correspondence is being transmitted via facsimile to the Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, facsimile number (703) 872-9306 on June 23, 2005.

By: _____
Michele Morrow

TRANSMITTAL DOCUMENT

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:
ENCLOSED HEREWITH:

- Appeal Brief (37 C.F.R. 41.37).

A fee of $500.00 is required for filing an Appeal Brief. Please charge this fee to IBM Corporation Deposit Account No. 09-0447. No additional fees are believed to be necessary. If, however, any additional fees are required, I authorize the Commissioner to charge these fees which may be required to IBM Corporation Deposit Account No. 09-0447. No extension of time is believed to be necessary. If, however, an extension of time is required, the extension is requested, and I authorize the Commissioner to charge any fees for this extension to IBM Corporation Deposit Account No. 09-0447.

Respectfully submitted,

Francis Lammes
Registration No. 55,353
Agent for Applicant

Duke W. Yee
Registration No. 34,285
Attorney for Applicant

YEE & ASSOCIATES, P.C.
P.O. Box 802333
Dallas, Texas 75380
(972) 385-8777

Docket No. AUS000185US1 *PATENT*

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re application of: **George** §
§
Serial No. **09/544,274** §
§
Filed: **April 6, 2000** §
§
For: **Apparatus and Method for Deletion of Objects from an Object-Relational System in a Customizable and Database Independent Manner** §

§ Group Art Unit: **2162**
§
§ Examiner: **Ly, Anh**

Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF (37 C.F.R. 41.37)

This brief is in furtherance of the Notice of Appeal, filed in this case on April 26, 2005.

The fees required under § 41.20(B)(2), and any required petition for extension of time for filing this brief and fees therefore, are dealt with in the accompanying TRANSMITTAL OF APPEAL BRIEF.

REAL PARTY IN INTEREST

The real party in interest in this appeal is the following party: International Business Machines Corporation.

RELATED APPEALS AND INTERFERENCES

With respect to other appeals or interferences that will directly affect, or be directly affected by, or have a bearing on the Board's decision in the pending appeal, there are no such appeals or interferences.

STATUS OF CLAIMS

A. TOTAL NUMBER OF CLAIMS IN APPLICATION

Claims in the application are: 1, 3-12, and 14-48.

B. STATUS OF ALL THE CLAIMS IN APPLICATION

1. Claims canceled: 2 and 13.
2. Claims withdrawn from consideration but not canceled: NONE
3. Claims pending: 1, 3-12, and 14-48.
4. Claims allowed: NONE
5. Claims rejected: 1, 3-12, and 14-48.
6. Claims objected to: NONE

C. CLAIMS ON APPEAL

The claims on appeal are: 1, 3-12, and 14-48.

STATUS OF AMENDMENTS

There are no amendments after the final rejection.

SUMMARY OF CLAIMED SUBJECT MATTER

Independent claims 1, 12, and 43:

The present invention provides a method of deleting object data from a relational database. (Specification, page 29, lines 2-5) The present invention determines a structure of the relational database, wherein determining the structure of the relational database includes referring to a database meta-information class object associated with the relational database. (Specification, page 29, lines 5-9) The present invention determines a delete action based on the structure of the relational database as described in the meta-information class object. (Specification, page 29, lines 10-12) The present invention generates database modification commands based on the determined delete action. (Specification, page 29, lines 12-15) The present invention sends the database modification commands to a relational database server, wherein the relational database server deletes the object data from the relational database based on the database modification commands. (Specification, page 29, lines 15-21)

The system recited in claim 12, as well as dependent claims 14-19, may be a bus system comprised of system bus **206**; I/O bus **212** or PCI buses **216**, **226**, and **228**; communication unit comprised of modem **218** or network adapter **220**, memory comprised of local memory **209**, processing unit comprised of processor **202** or processor **204** performing the steps described in the specification at page 29, lines 2-21, or equivalent. A person having ordinary skill in the art would be able to derive computer instructions on a computer readable medium as recited in claim 43, as well as dependent claims 44 and 45, given **Figure 10** and the corresponding description at page 29, lines 2-21, without undue experimentation.

Independent claims 20, 27, and 35:

The present invention provides a method of generating a class for deletion of data representations of objects in a relational database. (Specification, page 28, lines 18-20) The present invention determines a structure of the relational database. (Specification, page 28, lines 23-25) The present invention determines one or more delete actions based on the structure of the

relational database. (Specification, page 28, lines 26-30) The present invention generates a class object based on the determined structure and the determined one or more delete actions. (Specification, page 28, line 32 to page 29, line 1)

The means recited in independent claim 27, as well as dependent claims 28-34, may be data processing hardware within server **104** or clients **108**, **110**, and **112** in **Figure 1** operating under control of software performing the steps described in the specification at page 28, line 18, to page 29, line 1, or equivalent. A person having ordinary skill in the art would be able to derive computer instructions on a computer readable medium as recited in claim 35, as well as dependent claims 36-42, given **Figure 9** and the corresponding description at page 28, line 18, to page 29, line 1, without undue experimentation.

Independent claim 46:

The present invention provides a method of generating a class for deletion of data representations of objects in a relational database. (Specification, page 28, lines 18-21) The present invention determines a structure of the relational database. (Specification, page 28, lines 23-25) The present invention determines one or more default delete actions based on the structure of the relational database. (Specification, page 28, lines 26-30) The present invention receives user input to modify the one or more default delete actions. (Specification, page 28, lines 30-32) The present invention generates a class object based on the determined structure, the determined one or more delete actions and the user input. (Specification, page 28, line 32 to page 29, line 1)

GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

A. GROUND OF REJECTION (Claims 1, 3-4, 7-12, 14-15, and 17-19)

Claims 1, 3-4, 7-12, 14-15, and 17-19 are rejected under 35 U.S.C. § 103(a) as being allegedly unpatentable over Ng et al. (U.S. Patent No. 6,385,618 B1) in view of Text Book: Fundamentals of Database System (Third Edition) of Ramez Elmasri and Shamkant B. Navathe from Addison-Wisley and further in view of Sarkar (U.S. Patent 6,418,448 B1).

B. GROUND OF REJECTION (Claims 5, 6, 16, and 45)

Claims 5, 6, 16, and 45 are rejected under 35 U.S.C. § 103(a) as being allegedly unpatentable over Ng et al. (U.S. Patent No. 6,385,618 B1) in view of Text Book: Fundamentals of Database System (Third Edition) of Ramez Elmasri and Shamkant B. Navathe from Addison-Wisley and further in view of Sarkar (U.S. Patent 6,418,448 B1) and Crus et al. (U.S. Patent No. 4,947,320).

C. GROUND OF REJECTION (Claims 20, 21, 24-28, 31-36, 39-43, and 46-48)

Claims 20, 21, 24-28, 31-36, 39-43, and 46-48 are rejected under 35 U.S.C. § 103(a) as being allegedly unpatentable over Ng et al. (U.S. Patent No. 6,385,618 B1) in view of Text Book: Fundamentals of Database System (Third Edition) of Ramez Elmasri and Shamkant B. Navathe from Addison-Wisley.

D. GROUND OF REJECTION (Claims 22-23, 29-30, and 37-38)

Claims 22-23, 29-30, and 37-38 are rejected under 35 U.S.C. § 103(a) as being allegedly unpatentable over Ng et al. (U.S. Patent No. 6,385,618 B1) in view of Text Book: Fundamentals of Database System (Third Edition) of Ramez Elmasri and Shamkant B. Navathe from Addison-Wisley and further in view of Crus et al. (U.S. Patent No. 4,947,320).

E. GROUND OF REJECTION (Claim 44)

Claim 44 is rejected under 35 U.S.C. § 103(a) as being allegedly unpatentable over Ng et al. (U.S. Patent No. 6,385,618 B1) in view of Text Book: Fundamentals of Database System (Third Edition) of Ramez Elmasri and Shamkant B. Navathe from Addison-Wisley and further in view of Sarkar (U.S. Patent 6,418,448 B1).

ARGUMENT

A. GROUND OF REJECTION (Claims 1, 3-4, 7-12, 14-15, and 17-19)

The Office Action rejects claims 1, 3-4, 7-12, 14-15, and 17-19 under 35 U.S.C. § 103(a) as being allegedly unpatentable over Ng et al. (U.S. Patent No. 6,385,618 B1) in view of Text Book: Fundamentals of Database System (Third Edition) of Ramez Elmasri and Shamkant B. Navathe from Addison-Wisley and further in view of Sarkar (U.S. Patent 6,418,448 B1). This rejection is respectfully traversed.

As to independent claim 1, the Office Action states:

> With respect to claim 1, Ng discloses determining a structure of the relational database (database schema of a relational database: col. 4, lines 23-27 and lines 35-36), wherein determining the structure of the relational database includes referring to a database meta-information class object associated with the relational database (database metadata where information of data concerning data, data definition, characteristics, relationships and external data a database of a database management system: see abstract, col. 7, lines 60-67 and col. 8, lines 1-18; also see fig. 9); and structure of the relational database as described in the meta-information class object (see fig. 5, the object-relational mapping tool is import database schema, which is containing the information of relationship objects and the class object of the database: col. 6, lines 3-67 and col. 7, lines 12-20).
>
> Ng discloses structure of relational database and schemas of relational database. Ng does not explicitly indicate determining a delete action based on the structure of the relational database and generating database modification commands based on the determined delete action and sending the database modification commands. Elmasri-Navathe discloses active database rules and triggers as referred to as the Event-Condition-Action or ECA-model for the delete operation such as a cascade deletion, the organization or structure of the tables have to be determine to in order to delete tuple that reference the tuple that is being deleted (see rule R4, TOTALSAL4 (page 737 and page 210). In combination, Ng and Elmasri-Navathe do not teach the relational database server in Java via JDBC interface.
>
> However, Sarkar discloses java classes are loaded in the database server (col. 11, lines 45-55; also see col. 6, lines 7-22).
>
> Therefore, it would have been obvious to a person of ordinary skill in the art at the time the invention was made to combine the teachings of Ng in view of Elmasri-Navathe with the teachings of Sarkar so as to obtain database server of a object relational database locating of elements inside component relational schema with Java classes (col. 6, lines 13-15). This combination would provide a

> relational database having database server in the Java classes as argument for the interface of JDBC with SQL in the multi-tier client/server environment (Sarkar – col. 6, lines 20-28) and it is carrying an object SQL query for execution within one or more object relational schema (Sarkar – col. 6, lines 58-65 and querying and viewing multiple object relational schema in the large existing database (Sarkar – col. 7, lines 10-14) in the deletion of object in the relational database environment.

Office Action dated August 19, 2004, pages 3-5.

Claim 1, which is representative of the other rejected independent claim 12 with regard to similarly recited subject matter, reads as follows:

> 1. A method of deleting object data from a relational database, comprising:
> determining a structure of the relational database, wherein determining the structure of the relational database includes referring to a database meta-information class object associated with the relational database;
> determining a delete action based on the structure of the relational database as described in the meta-information class object;
> generating database modification commands based on the determined delete action; and
> sending the database modification commands to a relational database server, wherein the relational database server deletes the object data from the relational database based on the database modification commands. (emphasis added)

Appellant respectfully submits that Ng and Sarkar, taken alone or in combination, fail to teach or suggest referring to a database meta-information class object to determine the structure of a database and then determine a delete action based on the structure of the database determined from the meta-information class object, as recited in claims 1 and 12.

Ng teaches a system for updating an original object model, possibly having customizations from a programmer, with only the changes to the database made by a database administrator, as represented in the new database data structure. The result of the update of the original object model is a combination of the original object model, any customizations added by the programmer, and the changes to the database made by the administrator. Once the original object model is updated, new source code is generated.

Sarkar teaches a method and apparatus for processing markup language specifications for data and metadata used inside multiple related internet documents. The method and apparatus of

Sarkar are used to navigate queries and manipulate information from a plurality of object relational databases over the World Wide Web.

Elmasri teaches that a delete operation may fail when the structure of a relational database is such that an entry that is being deleted is referenced by entries in other tables of the relational database. In such a case, the deletion may be rejected, a cascade delete operation may be attempted or the referencing attribute values that cause the failure may be modified (see page 210, section 7.3.2).

The Final Office Action dated January 27, 2005, states:

> Ng. Et al. Of 6,385,618 (hereinafter Ng) teaches object relational mapping tools read database schema information and automatically generating a number of class objects whose inter-relationship to the structure of the database or schema of the database in the Java environment (col. 2, lines 12-24). Also teaches modifications or update or delete of the class object in the object relational database col. 2, lines 35-53). Also Sarkar of 6,418,448 teaches insert, delete and update a transaction of a database by using SQL statement (see fig. 18 and col. 21, lines 7-28).

Column 2, lines 12-53, read as follows:

> Object-relational mapping tools read database schema information and automatically generate source code from the database. This source code contains a number of classes whose interrelationships reflect the logical structure, or schema, of the database. A class, such as a Java.TM. class, is a data structure containing both data members that store data and function members (or methods) that act upon the data. The source code may contain one class for each table in the database, and each class may have a data member for each column in the corresponding table. Additionally, the classes contain function members that are used to both get and set the values for the data members and, eventually, update the database.
>
> By using an object-relational mapping tool, a programmer can automatically generate source code to facilitate their database application development. After the source code is generated, the programmer writes code to interact with only the classes in the source code and not the database, thus hiding the complexities of interacting with the database from the programmer. This allows a programmer who is familiar with object-oriented programming to code against familiar classes and not unfamiliar, sometimes cumbersome to use, database query languages.
>
> Although beneficial to programmers, conventional object-relational mapping tools suffer from a limitation. When a programmer runs the object-relational mapping tool, it generates source code with classes that reflect the structure of the database at that time. However, during the lifetime of the database, it is common for a database administrator to change the schema of the

> database (e.g., add a new field or table). Likewise, it is common for the programmer to update the classes in the source code (e.g., change a field name or delete a field). As such, both the classes and the database tend to evolve and change over time. Conventional object-relational mapping tools, however, are of little help in such situations. These tools can only remap the database to generate classes that contain the database modifications, but which do not contain the programmer's modifications. Therefore, the programmer's changes are lost and must be manually recreated, thus wasting significant development time. It is therefore desirable to improve object-relational mapping tools. (emphasis added)

In this section, Ng teaches object-relational mapping tools that read a database schema and create source codes from the database. This section fails to teach or suggest determining a delete action based on a structure of the database or generating database modification commands based on the determined delete action. There simply is no mention anywhere in Ng to determine a delete action based on a structure of the database or to generate database modification commands based on the determined delete action.

The Final Office Action dated October 7, 2003, alleges that these features are taught at column 4, lines 14-38. The cited section is included in the column 4, lines 14-54, which reads as follows:

> Methods and systems consistent with the present invention provide an improved object-relational mapping tool that integrates both customizations to source code and modifications to a database. Accordingly, the programmer does not have to recreate their customizations to the source code when the database changes, thus saving significant development time over conventional systems.
> Overview
>
> In accordance with methods and systems consistent with the present invention, the improved object-relational mapping tool maps a database by first querying the database to determine its schema and then by creating an internal data structure (known as the "database data structure") representing that schema. From this data structure, the object-relational mapping tool creates an object model containing all of the information necessary to generate classes and then creates source code containing a number of Java classes that may be used by a programmer to interface with the database.
>
> At some point during the lifetime of the object model, the programmer may add customizations to the object model (e.g., rename a field) that will be reflected in the source code, and the database administrator may likewise make a change to the database schema (e.g., add a column). After the database schema has been changed, the programmer may wish to update their source code to reflect the schema change while maintaining their customizations. To accomplish this goal, the object-relational mapping tool, in accordance with methods and systems consistent with the present invention, imports the new schema, creates a new

> database data structure, and compares the original (or preexisting) database data structure with this newly created one, noting any differences. Having noted the differences, the object-relational mapping tool has isolated the changes to the schema, and it then incorporates these changes into the existing object model and generates new source code. As a result, both the programmer's customizations to the object model (reflected in the old version of the source code) as well as the changes to the schema made by the database administrator are integrated into the new source code, thus saving the programmer significant development time over conventional systems. (emphasis added)

This section of Ng teaches that the programmer can make changes to the object model and the database administrator can make changes to the database schema, both of which can be isolated by the object-relational mapping tool, incorporated into the existing object model and generated into new source code. However, nowhere in this section, or any other section of Ng, is it taught or suggested that this process involves determining a delete action based on the schema or generating database modification commands based on the determined delete action and sending the database modification commands to a relational database server.

The Office Action dated July 16, 2003, acknowledges that Ng does not teach the limitation of the relational database server deleting the object data from the relational database based on the database modification commands, but that Sarkar allegedly teaches this feature. However, Sarkar does not provide for the deficiencies of Ng. That is, Sarkar does not teach determining a delete action based on a determined structure of a relational database or generating database modification commands based on a determined delete action.

While Sarkar may teach loading Java classes in a database server as alleged by the Office Action, there is no teaching or suggestion in Sarkar regarding determining a delete action based on the structure of a relational database, generating database modification commands based on the determined delete action, or sending the database modification commands to the relational database server. Loading of classes in a database server is not equivalent to sending database modification commands that are generated based on a delete action determined based on a determined structure of a relational database.

Furthermore Sarkar is directed to solving a completely different problem than that of Ng. Ng is directed to preserving object model customizations while accommodating changes to database schema. Sarkar is directed to a system for navigating, querying and manipulating information from a plurality of object relational databases over the Internet. One of ordinary

skill in the art would not have been motivated to combine these two systems at least because they are directed to two disparate fields of technology.

None of the references teaches or suggests referring to a database meta-information class object to determine the structure of a database and then determine a delete action based on the structure of the database determined from the meta-information class object, as recited in claims 1 and 12. While Ng teaches using a DatabaseMetaData interface of JDBC to obtain database schema information and storing that database schema information in a data structure, such as data structure 700 in Figure 7 of Ng, there is no teaching or suggestion in Ng that a delete action is determined by obtaining structure information from the data structure and then determining a delete action based on the structure obtained from the data structure. To the contrary, Ng merely uses the database schema in the data structure to isolate changes between two database data structures, as shown in Figure 10 of Ng. This process includes determining if the number of tables has changed between database data structures, determining if the type, name or number of fields in the hash table of the two database data structures has changed, comparing primary keys for each table to determine if a different primary key has been designated as the primary key, and comparing foreign keys between both database data structures to determine if any of the foreign keys have changed. Based on these identified changes, an object model is updated for the relational database and source code is then generated based on the updated object model (see column 7, line 13 to column 8, line 38).

The Final Office Action dated January 27, 2005, states:

> Ng teaches the structure of a relational database and with some command for deleting or modification to the class object in that structure (see fig. 7 and col. 2, lines 12-54) and col. 3, lines 60-67 and col.4, lines 1-10).

These sections of Ng have been discussed in detail above. That is, Ng teaches that the programmer can make changes to the object model and the database administrator can make changes to the database schema, both of which can be isolated by the object-relational mapping tool, incorporated into the existing object model and generated into new source code and using a DatabaseMetaData interface of JDBC to obtain database schema information and storing that database schema information in a data structure. Ng does not teach or suggest referring to a database meta-information class object to determine the structure of a database and then determine a delete action based on the structure of the database determined from the meta-

information class object. Furthermore, the statement by the Examiner in the Final Office Action dated January 27, 2005, is completely contradictory to the acknowledgement by the Examiner in the July 7, 2003, Office Action that Ng does not teach the limitation of the relational database server deleting the object data from the relational database based on the database modification commands.

Nowhere in Ng is there any teaching or suggestion that the database data structures that store the database schema are used to determine a structure of the database and then to determine a delete action based on the structure of the database obtained from the database data structure. To the contrary, if an object is to be deleted from a table in the relational database of Ng, the delete operation will be performed in a conventional manner, such as that taught by Elmasri. That is, the delete action will be attempted and if it fails due to an integrity violation, the delete action may be rejected, a cascade delete operation may be attempted, or referencing attribute values that cause the integrity violation may be modified (see Elmasri, page 210, section 7.3.2). This delete action is not determined based on database structure information obtained from a meta-information class object. To the contrary, the delete actions in both Ng and Elmasri are based on attempting delete operations on the relational database itself and determining if the operation fails. Ng and Elmasri, taken alone or in combination, fail to teach or suggest using a database meta-information class object to determine a structure of a relational database and then determine a delete action based on the structure of the relational database determined using the meta-information class object.

Sarkar does not make up for these deficiencies either. While Sarkar may teach Java classes being loaded in a database server, Sarkar does not teach or suggest identifying a structure of a relational database by referring to a database meta-information class object associated with the relational database or determining a delete action based on the structure of the relational database determined from the database meta-information class object.

The Office Action alleges that the database meta-information class object is taught by Ng at column 7, lines 60-67, column 8, lines 1-18, and in Figure 9. These portions of the Ng reference are directed to the DatabaseMetaData interface of the Java Database Connectivity (JDBC) tool. While the DatabaseMetaData interface may be used to obtain database schema information, the DatabaseMetaData interface is a set of methods associated with the JDBC tools

and is not a database meta-information class object itself that is associated with a relational database. To the contrary, the DatabaseMetaData interface methods form a tool that may be used to generate a database data structure that stores the database schema. Furthermore, the DatabaseMetaData interface is not used in Ng to determine a structure of a relational database such that a delete action is determined based on the structure of the relational database determined via the DatabaseMetaData interface.

Thus, in view of the above, Appellant respectfully submits that Ng, Sarkar, and Elmasri, taken alone or in combination, fail to teach or suggest determining a structure of a relational database by referring to a database meta-information class object associated with the relational database and determining a delete action based on the determined structure of the relational database as described in the meta-information class object, as recited in claims 1 and 12. At least by virtue of their dependency on claims 1 and 12, the specific features of dependent claims 3-4, 7-11, 14, 15 and 17-19 are not taught or suggested by Ng, Sarkar, and Elmasri, either alone or in combination. Accordingly, Appellant respectfully requests that the rejection of claims 1, 3, 4, 7-12, 14, 15 and 17-19 under 35 U.S.C. § 103(a) not be sustained.

A.1. Claims 4 and 15

With regard to claims 4 and 15, Ng, Sarkar and Elmasri, taken alone or in combination, fail to teach or suggest that the database meta-information class object includes a delete action identifier for each dependent table of a plurality of tables in a relational database. The Office Action alleges that this feature is taught by Ng at column 3, lines 62-67, column 7, lines 60-67, column 8, lines 1-17, and in Figure 9 (see rejection of claims 4 and 7 on page 5 of the Office Action dated August 19, 2004). Column 3, lines 62-27, reads as follows:

> The secondary storage device contains a database having a logical structure comprising tables with rows and columns. The memory contains a first database data structure reflecting the logical structure of the database and the object model containing objects based on the first database data structure.

This portion of Ng merely states that the database has tables with rows and columns and that the memory contains a database data structure that reflects the logical structure of the database and the object model. There is nothing in this section of Ng that teaches or even suggests a meta-

information class object that includes a delete action identifier for each dependent table of a plurality of tables in a relational database.

Column 7, line 60 to column 8, line 17, of Ng, which describes Figure 9, of Ng, reads as follows:

> FIG. 9 depicts a flowchart of the states performed when importing the database schema. Below, the object-relational mapping tool utilizes a number of methods which are found on the DatabaseMetaData interface of JDBC. The first state performed by the object-relational mapping tool is to call the GetTable method of the JDBC interface, which returns a description of the tables of the database (state 902). After retrieving this table information, the object-relational mapping tool selects one of the tables (state 904) and invokes the GetColumns method on the JDBC interface, returning a description of all of the columns in that table (state 906). Next, the object-relational mapping tool invokes the GetPrimaryKeys method to receive the primary key for the table (state 908). After obtaining the primary key, the object-relational mapping tool invokes the GetImportedKeys method to obtain information regarding the foreign keys (state 910). After invoking this method, the object-relational mapping tool determines if there are additional tables to be processed (state 912). If so, processing continues to state 904. Otherwise, the object-relational mapping tool constructs a database data structure, like the one shown in FIG. 7, from all of the information received in the previous states (state 914).

Nowhere in this portion of Ng is there any teaching or suggestion regarding including a delete action identifier, for each dependent table of a plurality of tables in a relational database, in a meta-information class object associated with the relational database. To the contrary, all this section of Ng teaches is the use of the various methods made available in the DatabaseMetaData interface of the JDBC to determine the structure of the relational database and then to use this information to generate the data structure shown in Figure 7, which is reproduced below:

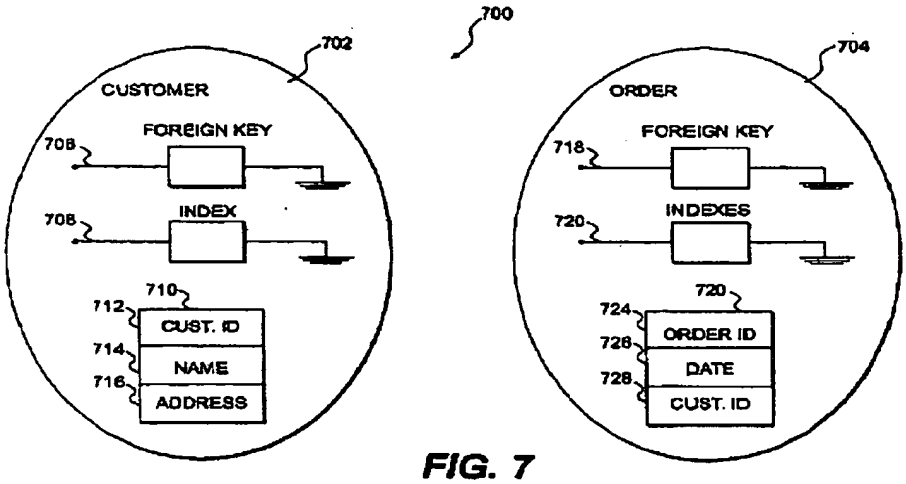


FIG. 7

Conspicuously missing from this data structure 700 is any delete identifier. To the contrary, as shown above, the only elements of this data structure are objects 702 and 704, relation objects 706, 708, 718, and hash tables 710 and 720. Nowhere in the data structure depicted in Figure 7 is there any delete action identifier, let alone a delete action identifier for each dependent table of a plurality of tables in a relational database.

Thus, despite the allegations made by the Office Action, Ng does not actually teach the feature of a database meta-information class object including a delete action identifier for each dependent table of a plurality of tables in a relational database. Furthermore, as stated above, neither of the other references, Sarkar and Elmasri, teaches or suggests this feature either. Sarkar has nothing to do with delete action identifiers in meta-information class objects and is merely used to allegedly teach Java classes being loaded into a database server. Elmasri, while teaching that a cascade delete operation may be attempting when a delete action fails due to the entry being deleted also being referenced by other entries in other tables of the relational database, provides no teaching or suggestion with regard to including a delete action identifier, for each dependent table of a plurality of tables in a relational database, in a meta-information class object that is associated with the relational database. Since none of these references alone teach or suggest this feature, any alleged combination of these references still would not result in this feature being taught or suggested.

Therefore, in addition to being dependent on independent claims 1 and 12, dependent claims 4 and 15 are also distinguishable over Ng, Sarkar, and Elmasri, either alone or in combination, by virtue of the specific features recited in these claims. Accordingly, Appellant respectfully requests that the rejection of claims 4 and 15 under 35 U.S.C. § 103(a) not be sustained.

A.2. Claims 7-10 and 17-19

Dependent claims 7-10 and 17-19 recite a file describing the structure and delete actions for tables in a relational database. These claims further define the file as being an Extensible Markup Language file, being generated based on user input to override default delete action identifiers in the file, and being generated based on user input to insert one or more delete constraints in the file for one or more of the tables in the relational database. None of these features are taught by Ng, Sarkar or Elmasri, either alone or in combination, because none of these references teach or suggest a file describing the structure and delete actions for tables in a relational database.

The Office Action alleges that these features are taught at column 3, lines 62-67, column 7, lines 60-67, column 8, lines 1-17, Figure 9, and column 7, lines 16-26 of Ng. Column 3, lines 62-67 of Ng reads as follows:

> The secondary storage device obtains a database having a logical structure comprising tables with rows and columns. The memory contains a first database data structure reflecting the logical structure of the database and an object model containing objects based on the first database data structure.

Nothing in this section of Ng teaches a file that describes the structure and delete actions for tables in a relational database. At most, the database data structure referenced in this section teaches a structure of the relational database. Nothing in Ng teaches a file that describes the structure and delete actions for tables in a relational database. The other sections of Ng reference by the Office Action merely describe the DatabaseMetaData interface of JDBC and the methods executed in the flow shown in Figure 9, which have been addressed in detail above. Moreover, since none of these sections of Ng teach or suggest a file that describes the structure and delete actions for tables in a relational database, Ng cannot teach that the file is an

Extensible Markup Language file, the file is generated based on user input to override default delete action identifiers in the file, or that the file is generated based on user input to insert one or more delete constraints in the file for one or more of the tables in the relational database. None of the references teaches or suggests these features.

Therefore, in addition to being dependent on independent claims 1 and 12, dependent claims 7-10 and 17-19 are also distinguishable over Ng, Sarkar, and Elmasri, either alone or in combination, by virtue of the specific features recited in these claims. Accordingly, Appellant respectfully requests that the rejection of claims 7-10 and 17-19 under 35 U.S.C. § 103(a) not be sustained.

A.3. Claim 16

With regard to claim 16, this claim recites that the delete action identifier is one of cascade delete and nullify columns delete. While the prior art teaches these two different types of delete operations, nowhere in the cited art is there any teaching of a delete action identifier in a database meta-information class object that is one of a cascade delete and a nullify columns delete identifier, as recited in claim 16. Simply teaching these delete operations does not make obvious a delete identifier in a meta-information class object that is one of a cascade delete and a nullify columns delete identifier.

Therefore, in addition to being dependent on independent claim 12, dependent claim 16 is also distinguishable over Ng, Sarkar, and Elmasri, either alone or in combination, by virtue of the specific features recited in these claims. Accordingly, Appellant respectfully requests that the rejection of claim 16 under 35 U.S.C. § 103(a) not be sustained.

B. GROUND OF REJECTION (Claims 5, 6, 16, and 45)

The Office Action rejects claims 5, 6, 16, and 45 under 35 U.S.C. § 103(a) as being allegedly unpatentable over Ng et al. (U.S. Patent No. 6,385,618 B1) in view of Text Book: Fundamentals of Database System (Third Edition) of Ramez Elmasri and Shamkant B. Navathe

from Addison-Wisley and further in view of Sarkar (U.S. Patent 6,418,448 B1) and Crus et al. (U.S. Patent No. 4,947,320).

This rejection is respectfully traversed for at least the same reasons as noted above with regard to claims 1, 12, and 43, which is similar in subject matter to claims 1 and 12, from which claims 5, 6, 16, and 45 depend. Specifically, Ng, Elmasri, and Sarkar, taken alone or in combination, fail to teach or suggest determining a structure of a relational database by referring to a database meta-information class object and determining a delete action based on the structure described in the meta-information class object, or a class object that is generated based on a structure of a relational database and one or more delete actions for tables in the relational database.

Moreover, Crus does not provide for the deficiencies of Ng, Elmasri, and Sarkar. Crus teaches a delete set null and a delete cascade operation, as discussed in previously filed Responses. However, Crus provides no teaching or suggestion regarding a class object that is generated based on a structure of a relational database and one or more delete actions for tables in the relational database. Crus also provides no teaching or suggestion regarding determining a structure of a relational database by involving a meta-information class object associated with the relational database and then determining a delete action based on the determined structure of the relational database. Thus, even if Crus were combinable with Ng, Elmasri, and Sarkar, the result still would not be the invention recited in independent claims 1, 12, and 43, from which claims 5, 6, 16, and 45 depend.

Furthermore, there is no teaching or suggestion in Crus to include a delete set null or a delete cascade operation identifier, for each dependent table of a plurality of tables in a relational database, in a meta-information class object, as recited in claims 5 and 16 or information identifying a delete set null or delete cascade operation in a class object, as recited in claim 45. While Crus may generally teach delete set null and delete cascade operations, there is nothing in Crus that teaches or suggests including information regarding such operations in a class object generated based on a structure of a relational database and one or more delete actions.

The Final Office Action dated January 27, 2005 states:

> Crus et al. Of 4,947,320 (hereinafter Crus) teaches the commands in the SQL such as Load, Insert, Update and Delete commands and their resulting operations. Especially, there are three rules of the delete operations: Delete

> Restrict, Delete Set Null and Delete Cascade (col. 5, lines 3-67, col.17, lines 1-67 and col. 18, lines 1-18) and delete operation is delete the tables in a relational datqabase (col. 24, lines 42-67 and col. 25, lines 1-40).

While Crus may generally teach delete set null and delete cascade operations and state using principles of relational databases, Crus does not teach or suggest determining the structure of the relational database. Thus, Crus fails to teach or suggest including information regarding such operations in a class object generated based on a structure of a relational database and one or more delete actions.

In view of the above, Appellant respectfully submits that none of the cited references, whether taken alone or in combination, teaches or suggests the features of independent claims 1, 12 and 43. At least by virtue of their dependency on claims 1, 12, and 43, the specific features of dependent claims 5, 6, 16, and 45 are not taught or suggested by the cited references, either alone or in combination. Accordingly, Appellant respectfully requests that the rejection of claims 5, 6, 16, and 45 under 35 U.S.C. § 103(a) not be sustained.

C. GROUND OF REJECTION (Claims 20, 21, 24-28, 31-36, 39-43, and 46-48)

The Office Action rejects claims 20, 21, 24-28, 31-36, 39-43, and 46-48 under 35 U.S.C. § 103(a) as being allegedly unpatentable over Ng et al. (U.S. Patent No. 6,385,618 B1) in view of Text Book: Fundamentals of Database System (Third Edition) of Ramez Elmasri and Shamkant B. Navathe from Addison-Wisley.

With regard claims 20, 27, and 35, these claims recite generating a class object based on a determined structure and determined one or more delete actions. Ng and Elmasri, taken alone or in combination, fail to teach or suggest this feature. While Ng uses the DatabaseMetaData interface of JDBC to obtain database schema information which is then stored in a database data structure, Ng does not generate this database data structure based on one or more delete actions determined based on the structure of the relational database. To the contrary, as shown in Figure 9 and described in columns 7 and 8, the DatabaseMetaData interface merely gets a description of tables in the database, gets a description of the columns in the tables, gets the primary keys for the tables, and gets the foreign keys for the tables. Nowhere in Ng is there any teaching or suggestion to use the DatabaseMetaData interface to determine one or more delete actions based

on the structure of a relational database and then generate a class object based on the determined one or more delete actions.

The Final Office Action dated January 27, 2005, states:

> Ng. teaches object relational mapping tools read database schema information and automatically generating a number of class objects whose inter-relationship to the structure of the database or schema of the database in the Java environment (col. 2, lines 12-24). Also teaches modifications or update or delete of the class object in the object relational database col. 2, lines 35-53 and the structure of a relational database and with some command for deleting or modification to the class object in that structure (see fig.7 and col. 2, lines 12-54) and col. 3, lines 60-67 and col. 4, lines 1-10).

As discussed above, these sections of Ng teach that the programmer can make changes to the object model and the database administrator can make changes to the database schema, both of which can be isolated by the object-relational mapping tool, incorporated into the existing object model and generated into new source code and using a DatabaseMetaData interface of JDBC to obtain database schema information and storing that database schema information in a data structure. Ng does not teach or suggest referring to a database meta-information class object to determine the structure of a database and then determine a delete action based on the structure of the database determined from the meta-information class object. Furthermore, Ng does not teach generating a class object based on a determined structure and determined one or more delete actions.

Elmasri does not teach or suggest these features either. Elmasri merely teaches attempting a cascade delete operation when a delete operation fails due to an integrity violation. Thus, Ng and Elmasri, taken alone or in combination, fail to teach or suggest the features of claims 20, 27, and 35. Therefore, since their dependent claims incorporate the subject matter of these respective independent claims, the dependent claims are also not taught or suggested by the alleged combination of Ng and Elmasri, either alone or in combination.

Regarding the remaining independent claims 43 and 46, these claims recite similar features to that emphasized above with regard to claims 20, 27, and 35. In particular, claim 43 recites a database meta-information generator class for generating a class object based on the determined structure and the determined one or more delete actions. Claim 46 recites generating a class object based on a determined structure, determined one or more delete actions, and user

input. Thus, for similar reasons as noted above with regard to claims 20, 27 and 35, claims 43 and 46 define over the proposed combination of Ng and Elmasri.

In view of the above, Appellant respectfully submits that Ng and Elmasri, taken alone or in combination, fail to teach or suggest the features of independent claims 20, 27, 35, 43, and 46. At least by virtue of their dependency, the specific features of dependent claims 21, 24-26, 28, 31-34, 36, 39-42, 47, and 48 are not taught or suggested by the alleged combination of Ng and Elmasri. Accordingly, Appellant respectfully requests that the rejection of claims 20, 21, 24-28, 31-36, 39-43, and 46-48 under 35 U.S.C. § 103(a) not be sustained.

C.1. Claim 24

Ng and Elmasri, either alone or in combination, do not teach or suggest the specific features recited in dependent claim 24. None of the cited references teach or suggest one or more delete actions being determined from a file describing the structure and delete actions for tables in the relational database, as recited in claim 24. The Office Action alleges that this feature is taught by Ng at the same portions discussed above and in previous responses. Again, there is nothing in Ng that teaches or suggests a file that describes the structure and delete actions for tables in a relational database. There is nothing in NG that teaches or suggests a data structure that identifies delete actions for tables in the relational database.

Therefore, in addition to being dependent on independent claim 20, dependent claim 24 is also distinguishable over Ng and Elmasri, either alone or in combination, by virtue of the specific features recited in these claims. Accordingly, Appellant respectfully requests that the rejection of claim 24 under 35 U.S.C. § 103(a) not be sustained.

C.2. Claims 47 and 48

Additionally, Ng and Elmasri, taken alone or in combination, fail to teach or suggest a user input that overrides one or more default delete actions (claim 47) or inserts one or more delete action constraints (claim 48). The Office Action alleges that these features are taught by Ng at column 4, lines 45-67, column 6, lines 42-64 and column 7, lines 9-67. Column 4, lines

45-67 merely teaches the incorporation of changes into an existing object model and the generation or source code. Column 6, lines 42-64 merely teaches methods in class 420 for getting and setting the values of data members and the use of a foreign key to create a relationship in source code. Column 7, lines 9-67 merely describes a process for a database administrator to add a column to a customer table. Nowhere in any of these sections, or any other section of Ng or Elmasri, is there any teaching or suggestion with regard to user input that overrides one or more default delete actions or inserts one or more delete action constraints.

Therefore, in addition to being dependent on independent claim 46, dependent claims 47 and 48 are also distinguishable over Ng and Elmasri, either alone or in combination, by virtue of the specific features recited in these claims. Accordingly, Appellant respectfully requests that the rejection of claims 47 and 48 under 35 U.S.C. § 103(a) not be sustained.

C.3. Claims 25-26, 31-34, and 39-42

Dependent claims 25-26, 31-34, and 39-42 recite a file describing the structure and delete actions for tables in a relational database. These claims further define the file as being an Extensible Markup Language file, being generated based on user input to override default delete action identifiers in the file, and being generated based on user input to insert one or more delete constraints in the file for one or more of the tables in the relational database. These features are not taught because none of the applied references teaches or suggests a file describing the structure and delete actions for tables in a relational database.

The Office Action dated August 19, 2004, alleges that these features are taught at column 3, lines 62-67, column 7, lines 60-67, column 8, lines 1-17, Figure 9, and column 7, lines 16-26 of Ng. Column 3, lines 62-67 of Ng reads as follows:

> The secondary storage device obtains a database having a logical structure comprising tables with rows and columns. The memory contains a first database data structure reflecting the logical structure of the database and an object model containing objects based on the first database data structure.

Nothing in this section of Ng teaches a file that describes the structure and delete actions for tables in a relational database. At most, the database data structure referenced in this section teaches a structure of the relational database. Nothing in Ng teaches a file that describes the

structure and delete actions for tables in a relational database. The other sections of Ng reference cited by the Office Action merely describe the DatabaseMetaData interface of JDBC and the methods executed in the flow shown in Figure 9, which have been addressed in detail above. Nowhere in any of these sections is there any teaching or suggestion of a file that describes the structure and delete actions for tables in a relational database. Moreover, since none of these sections of Ng teaches or suggests a file that describes the structure and delete actions for tables in a relational database, Ng cannot teach that the file is an Extensible Markup Language file, the file is generated based on user input to override default delete action identifiers in the file, or that the file is generated based on user input to insert one or more delete constraints in the file for one or more of the tables in the relational database. None of the references teaches or suggests these features.

Therefore, in addition to being dependent on independent claims 20, 27, and 35, dependent claims 25-26, 31-34, and 39-42 are also distinguishable over Ng and Elmasri, either alone or in combination, by virtue of the specific features recited in these claims. Accordingly, Appellant respectfully requests that the rejection of claims 25-26, 31-34, and 39-42 under 35 U.S.C. § 103(a) not be sustained.

D. GROUND OF REJECTION (Claims 22-23, 29-30, and 37-38)

The Office Action rejects claims 22-23, 29-30, and 37-38 under 35 U.S.C. § 103(a) as being allegedly unpatentable over Ng et al. (U.S. Patent No. 6,385,618 B1) in view of Text Book: Fundamentals of Database System (Third Edition) of Ramez Elmasri and Shamkant B. Navathe from Addison-Wisley and further in view of Crus et al. (U.S. Patent No. 4,947,320).

Appellant respectfully submits that the rejection is respectfully traversed for at least the same reasons as noted above with regard to claims 20, 27, and 35 from which claims 22-23, 29-30, and 37-38 depend. Specifically, Ng and Elmasri, taken alone or in combination, fail to teach or suggest determining a structure of a relational database by referring to a database meta-information class object and determining a delete action based on the structure described in the meta-information class object, or a class object that is generated based on a structure of a relational database and one or more delete actions for tables in the relational database.

Moreover, Crus does not provide for the deficiencies of Ng and Elmasri. Crus teaches a delete set null and a delete cascade operation, as discussed in previously filed Responses. However, Crus provides no teaching or suggestion regarding a class object that is generated based on a structure of a relational database and one or more delete actions for tables in the relational database. Crus also provides no teaching or suggestion regarding determining a structure of a relational database by involving a meta-information class object associated with the relational database and then determining a delete action based on the determined structure of the relational database. Thus, even if Crus were combinable with Ng and Elmasri, the result still would not be the invention recited in independent claims 20, 27 and 35, from which claims 22-23, 29-30 and 37-38 depend.

Furthermore, there is no teaching or suggestion in Crus to include information to identify a delete set null or delete cascade operation in a class object, as recited in claims 22, 29 and 37. While Crus may generally teach delete set null and delete cascade operations, there is nothing in Crus that teaches or suggests including information regarding such operations in a class object generated based on a structure of a relational database and one or more delete actions.

In view of the above Appellant respectfully submits that the cited references, whether taken alone or in combination, fail to teach or suggest the features of independent claims 20, 27, and 35. At least by virtue of their dependency on claims 20, 27, and 35, the specific features of dependent claims 22-23, 29-30, and 37-38 are not taught or suggested by the cited references, either alone or in combination. Accordingly, Appellant respectfully requests that the rejection of claims 22-23, 29-30, and 37-38 not be sustained.

E. GROUND OF REJECTION (Claim 44)

The Office Action rejects claim 44 under 35 U.S.C. § 103(a) as being allegedly unpatentable over Ng et al. (U.S. Patent No. 6,385,618 B1) in view of Text Book: Fundamentals of Database System (Third Edition) of Ramez Elmasri and Shamkant B. Navathe from Addison-Wisley and further in view of Sarkar (U.S. Patent 6,418,448 B1).

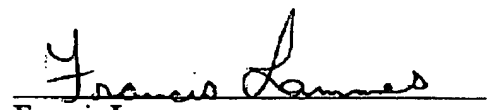
Appellant respectfully submits that the features of claim 44 are similar to features previously discussed above. That is, claim 44 recites that the database meta-information

generator class encapsulates information identifying a structure of a relational database and one or more delete actions into a class object. None of the cited art teaches or suggests a class object that encapsulates information identifying a structure of a relational database and one or more delete actions, as discussed at length above.

Therefore, in addition to being dependent on independent claim 43, dependent claim 44 is also distinguishable over Ng, Sarkar, and Elmasri, either alone or in combination by virtue of the specific features recited in these claims. Accordingly, Appellant respectfully requests that the rejection of claim 44 under 35 U.S.C. § 103(a) not be sustained.

CONCLUSION

In view of the above, Appellant respectfully submits that claims 1, 3-12, and 14-48 are allowable over the cited prior art and that the application is in condition for allowance. Accordingly, Appellant respectfully requests the Board of Patent Appeals and Interferences to not sustain the rejections set forth in the Final Office Action.



Francis Lammes
Reg. No. 55,353
YEE & ASSOCIATES, P.C.
PO Box 802333
Dallas, TX 75380
(972) 385-8777

CLAIMS APPENDIX

The text of the claims involved in the appeal are:

1. A method of deleting object data from a relational database, comprising:

determining a structure of the relational database, wherein determining the structure of the relational database includes referring to a database meta-information class object associated with the relational database;

determining a delete action based on the structure of the relational database as described in the meta-information class object;

generating database modification commands based on the determined delete action; and

sending the database modification commands to a relational database server, wherein the relational database server deletes the object data from the relational database based on the database modification commands.

3. The method of claim 1, wherein the database meta-information class object encapsulates a dependency structure of the relational database.

4. The method of claim 3, wherein the database meta-information class object further includes a delete action identifier for each dependent table of a plurality of tables in the relational database

5. The method of claim 4, wherein the delete action identifier is one of cascade delete and nullify columns delete.

6. The method of claim 1, wherein the delete action is one of cascade delete and nullify columns delete.

7. The method of claim 1, wherein the database meta-information class object is generated based on a file describing the structure and delete actions for tables in the relational database.

8. The method of claim 7, wherein the file is an Extended Markup Language file.

9. The method of claim 7, wherein the file is further generated based on user input to override default delete action identifiers in the file.

10. The method of claim 7, wherein the file is further generated based on user input to insert one or more delete constraints in the file for one or more of the tables in the relational database.

11. The method of claim 1, wherein the database modification commands are Structured Query Language (SQL) statements.

12. A data processing system for deleting object data from a relational database, comprising:

a data processor; and

a relational database storage device, wherein the data processor determines a structure of the relational database, wherein the data processor determines the structure of the relational database by referring to a database meta-information class object associated with the relational database, determines a delete action based on the structure of the relational database as described

in the meta-information class object, generates database modification commands based on the determined delete action and sends the database modification commands to the relational database storage device, wherein the relational database storage device deletes the object data from the relational database based on the database modification commands.

14. The data processing system of claim 12, wherein the database meta-information class object encapsulates a dependency structure of the relational database.

15. The data processing system of claim 14, wherein the database meta-information class object further includes a delete action identifier for each table of a plurality of tables in the relational database.

16. The data processing system of claim 15, wherein the delete action identifier is one of cascade delete and nullify columns delete.

17. The data processing system of claim 12, wherein the database meta-information class object is generated based on a file describing the structure and delete actions for tables in the relational database.

18. The data processing system of claim 17, further comprising a file editor application executed by the data processor, wherein the file editor application changes the delete action in the file for one or more of the tables in the relational database based on a user input to override default delete action identifiers in the file.

19. The data processing system of claim 18, wherein the file editor application inserts one or more delete constraints into the file for one or more of the tables in the relational database, based on a user input.

20. A method of generating a class for deletion of data representations of objects in a relational database, comprising:

determining a structure of the relational database;

determining one or more delete actions based on the structure of the relational database; and

generating a class object based on the determined structure and the determined one or more delete actions.

21. The method of claim 20, wherein generating the class object includes encapsulating information identifying the structure of the relational database and the one or more delete actions.

22. The method of claim 21, wherein the one or more delete actions is at least one of cascade delete and nullify columns delete.

23. The method of claim 20, wherein the one or more delete actions is at least one of cascade delete and nullify columns delete.

24. The method of claim 20, wherein the structure of the relational database and the one or more delete actions are determined from a file describing the structure and delete actions for tables in the relational database.

25. The method of claim 24, wherein the file is further generated based on user input to override default delete action identifiers in the file.

26. The method of claim 24, wherein the file is further generated based on user input to insert one or more delete constraints in the file.

27. An apparatus for generating a class object for deletion of data representations of objects in a relational database, comprising:

means for determining a structure of the relational database;

means for determining one or more delete actions based on the structure of the relational database; and

means for generating the class object based on the determined structure and the determined one or more delete actions.

28. The apparatus of claim 27, wherein the means for generating the class object encapsulates information identifying the structure of the relational database and the one or more delete actions.

29. The apparatus of claim 28, wherein the one or more delete actions is at least one of cascade delete and nullify columns delete.

30. The apparatus of claim 27, wherein the one or more delete actions is at least one of cascade delete and nullify columns delete.

31. The apparatus of claim 27, wherein the means for determining the structure of the relational database and the means for determining the one or more delete actions determine the structure and one or more delete actions from a file describing the structure and delete actions of tables in the relational database.

32. The apparatus of claim 31, further comprising means for generating the file, wherein the file is generated based on Java Database Connectivity (JDBC) database metadata associated with the relational database.

33. The apparatus of claim 31, wherein the file is further generated based on user input to override default delete action identifiers in the file.

34. The apparatus of claim 31, wherein the file is further generated based on user input to insert one or more delete constraints in the file.

35. A computer program product in a computer readable medium for generating a class object for deletion of data representations of objects in a relational database, comprising:

first instructions for determining a structure of the relational database;

second instructions for determining one or more delete actions based on the structure of the relational database; and

third instructions for generating the class object based on the determined structure and the determined one or more delete actions.

36. The computer program product of claim 35, wherein the third instructions include instructions for encapsulating information identifying the structure of the relational database and the one or more delete actions.

37. The computer program product of claim 36, wherein the one or more delete actions is at least one of cascade delete and nullify columns delete.

38. The computer program product of claim 35, wherein the one or more delete actions is at least one of cascade delete and nullify columns delete.

39. The computer program product of claim 35, wherein the first and second instructions determine the structure of the relational database and the one or more delete actions from a file describing the structure and delete actions for tables in the relational database.

40. The computer program product of claim 39, further comprising fourth instructions for generating the file based on Java Database Connectivity (JDBC) database metadata associated with the relational database.

41. The computer program product of claim 39, wherein the fourth instructions further include instructions for generating the file based on user input to override default delete action identifiers in the file.

42. The computer program product of claim 39, wherein the fourth instructions further include instructions for generating the file based on user input to insert delete action constraints in the file.

43. A computer program product in a computer readable medium for generating a class object for deletion of data representations of objects in a relational database, comprising:

a meta-information class for determining a structure of the relational database and one or more delete actions based on the structure of the relational database; and

a database meta-information generator class for generating the class object based on the determined structure and the determined one or more delete actions.

44. The computer program product of claim 43, wherein the database meta-information generator class encapsulates information identifying the structure of the relational database and the one or more delete actions into the class object.

45. The computer program product of claim 44, wherein the one or more delete actions is at least one of cascade delete and nullify columns delete.

46. A method of generating a class for deletion of data representations of objects in a relational database, comprising:

determining a structure of the relational database;

determining one or more default delete actions based on the structure of the relational database;

receiving user input to modify the one or more default delete actions; and

generating a class object based on the determined structure, the determined one or more delete actions and the user input.

47. The method of claim 46, wherein the user input overrides one or more of the one or more default delete actions.

48. The method of claim 46, wherein the user input inserts one or more delete action constraints.

EVIDENCE APPENDIX

There is no evidence to be presented.

RELATED PROCEEDINGS APPENDIX

There are no related proceedings.